Aalto University
School of Science
Degree Programme in Computer Science and Engineering

Julien Blégean

Twitter the Rioter :

Analyzing roles through a protest on social media.

What was your part during the 2014 Ferguson riots?

Final Project
Espoo, May 25, 2015

Supervisor Professor Aristides Gionis, Aalto University
Advisor Michael Mathioudakis, PhD

Aalto University
School of Science
Degree Programme in Computer Science and Engineering

ABSTRACT OF FINAL PROJECT

Author : Julien Blégean

Title : Twitter the Rioter : Analyzing roles through a protest on social media.
What was your part during the 2014 Ferguson riots?

Date : May 25, 2015

Number of pages : 68 **Language** : English

Supervisor : Professor Aristides Gionis

Advisor : Michael Mathioudakis, PhD

Abstract :

We introduce a framework for analyzing roles of users during a riot. The dataset contains messages from Twitter during the 2014 US Ferguson protests.

First, after some preprocessing one the data, we extract topics from a riot by comparing two techniques : k-means and Latent Dirichlet Allocation.

Secondly, we focus on the content of the tweets. We train and test a Naive Bayes classifier to predict if a tweet is either supportive or informative about the riot. We also study the medias shared by users in order to improve our classifier and in the end we define and compute user polarity score.

Then, we perform graph analysis on the hashtags and users and we visualize these networks through examples. We also define and compute user influence score with degree centrality and Page Rank.

Finally, we define the role of a user during a riot (or several) as being the feature vector of its polarity and influence scores for each topic. We perform dimension reduction with PCA and t-SNE for visualization purpose and neighbors research. Through experiments, we show how users with the same job (for example journalists or activists) are grouped together.

Keywords : Twitter, Natural Language Processing, Graph Analysis, Visualization, Naive Bayes, Clustering, Topic Modeling, Influence, Social Network, Riot, Protest, Ferguson

Acknowledgments

First and foremost, I would like to particularly thank my supervisor, Prof. Aristides Gionis and my advisor, Dr. Michael Mathioudakis, for their guidance and advices all through this project.

I also would like to thank the data mining group for welcoming me during these 5 months.

More generally, I would like to thank to all the wonderful people I have met during my exchange year in Finland, for their support and friendship.

Finally, *merci* to my family and friends in France for their support, three thousand kilometers from here.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

London, 2011. Thousands of people riot in the capital and in many cities across England between 6th and 11th August. Twitter, the 500-million-users social network, is rapidly accused of being the tool prompting looting and violence in the streets. The same year, indeed, The Daily Mail writes : "There was concern that the disturbances were fanned by Twitter, with some of those taking part posting inflammatory comments from the scene and calling for reinforcements."[1]

According to the Guardian[2], Prime Minister David Cameron, asked companies such as Twitter to take responsability for the content posted on their websites. Moreover, as for the BBC[3], the British Government even considered to shut down these social networks temporary in order to avoid their use in the riots.

For the first time, London riots highlighted the possible use of social medias to commit violence. At least, the British government enhanced the capacity of the network to organize protests, which was probably not expected when Twitter was created in 2006, only 5 years before London events.

But is Twitter really a "rioter" ? How to understand this new use of social medias ? According to Wikipedia[4], 14 riots occured worldwide. In 2012, there were 45, and in 2011, more than 100. Besides, more people get accustomed to technologies and social medias every day, and are ready to use them in one way or another during a crisis event, such as a riot.

The question of being able to read, analyze and extract information from these networks during a protest becomes central.

[1] Read more: `http://www.dailymail.co.uk/news/article-2023254/Tottenham-riot-Mark-Duggan-shooting-sparked-police-beating-girl.html`
[2] `http://www.theguardian.com/uk/2011/aug/11/cameron-call-social-media-clampdown`
[3] `http://www.bbc.com/news/technology-14493497`
[4] `http://en.wikipedia.org/wiki/List_of_riots`

1.2 2014 Ferguson unrest

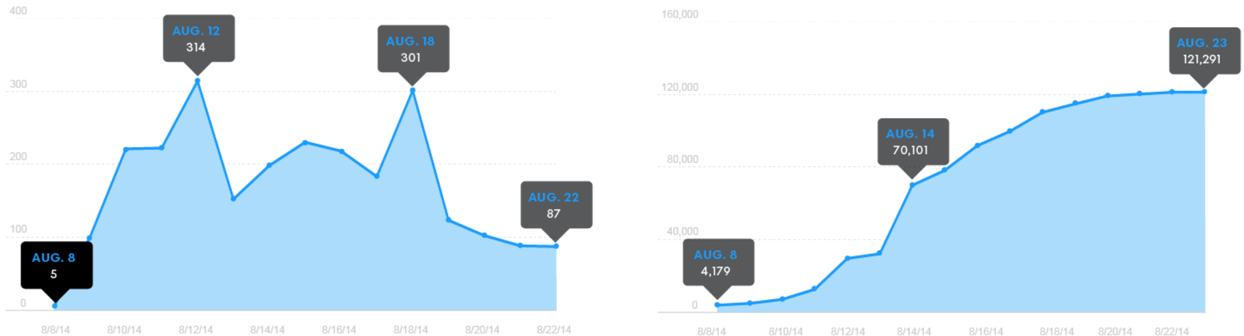
On August 9th, 2014, a 18-year-old black man called Michael Brown is shot dead in Ferguson by a white police officer. This event was the trigger of riots, both peaceful and violent, for more than 2 weeks.

On November 24th, the Grand Jury decides not to indict the police officier who shot dead Michael Brown, sparking unrest in Ferguson and in others cities accross the US, like in Boston or Los Angeles.

As for former unrests, the Ferguson protest has been widely broadcasted on social media like Twitter.

Figure 1.1: Photo of the Ferguson riots. On the left, policemen, on the right, protesters raising hands and saying ”Don’t Shoot”

Some people, like Antonio French, were particularly active. Although not being a journalist, but leaving near Ferguson, he reported daily through Twitter what was happening in the field.



(a) Tweets count per day

(b) Followers cumulative count per day

Figure 1.2: Antonio French’s activity during the protest (© `usatoday.com`)

One can see on the figure 1.2 how Antonio French was active (up to 300 tweets a day) during the whole unrest. And, even more interesting, the figure shows how he became popular, as he was followed by more than 120,000 people at the end of the protest, whereas he was almost unknown on Twitter before.

As the article shows through example tweets, Antonio French had an informative role. He mostly posted pictures and videos during each riot, and everytime, he tried to gather information about what was happening.
In this way, it is interesting to wonder:

* How to model such a role on Twitter during a riot ?
* How to say if two users are similar ?
* How to group users given the role they play ?
* How to visualize this kind of information ?

The following part describes the problem and the objectives more formally.

1.3 Formulation and objectives

In this part we introduce clearly the objectives this project will try to fulfil.
Firstly, we define and formulate three entities : a **protest**, a **riot** and a **topic**.

Definition - Protest $\mathcal{P}_n$
A protest $\mathcal{P}_m$ is defined as a time ordered list of m riots r.

$$\mathcal{P}_m = [r_1, r_2, \cdots, r_m]$$

In our case, we suppose there is a riot each night. As a consequence, $\mathcal{P}_m$ is m-days long.

Definition - Riot r and topics t
A riot r is defined as a set of n topics t occuring during the riot r.

$$r = [t_1, t_2, \cdots, t_n]$$

The main objective of this project is given by :

Main Objective : Define and compute the **role** of a user, the similarity between two users and clusters these users by their role.

We decide to characterize the role of a user by a simple model with two features :

* **Polarity**. What kind of content does the user publish ? How to characterize it ? What are the different types of content existing ? How to quantify it ?
* **Influence**. Does the user has influence ? How to quantify it? How to visualize it ? How does this influence evolve ?

This leads to two intermediary objectives :

Objective 1. Define and compute the **polarity** of a user.

Objective 2. Define and compute the **influence** of a user.

1.4 Technical details

All the code has been written in Python and uses the following libraries :

* **NLTK**, for the Natural Language Processing part ;
* **Scikit-Learn**, for the machine learning algorithms ;
* **Gensim**, for the topic modeling part ;
* **Networkx**, for creating and manipulating graphs ;

The code is hosted on Github at the following url :

`http://github.com/jwheatp/twitter-riots`

The computation tasks have been made on the Linux Ubuntu `brute.aalto.fi` server, model HP BL460c.

Chapter 2

Dataset

2.1 Origin

The choice of the Ferguson riots as a case of study has been made for several reasons. Firstly, this is a recent case, as it occured last year. Moreover, this events happened in a country that use social media a lot. In addition, the fact all the tweets are in English makes the study of the language easier than studying, for example, the Arab Spring protests. But one of the main reason was that this dataset is available.

Indeed, obtaining datasets about riots is not a simple task. Twitter does not allow to fetch past tweets between two dates with a precise keyword, but only new tweets from the Stream API. Some research groups manage to grant a special access from Twitter, like for the London riots study[Procter et al., 2013], but do not make these datasets publicly available online.

Moreover, Twitter does not allow neither peers to share full datasets obtained with its API. However, the company allows people for now to share IDs list of tweets, without any content. One is then able to use the Twitter API to fetch, for each ID, the tweet content (text, author, date etc.) under the rates limitations.

The dataset used in this project is obtained from Ed Summers, a developer who started to record the Ferguson-related tweets after Mike Brown's shooting and who published on his blog the IDs list[1]. The task was then to *hydrate* the tweets and users to obtain the full dataset.

Contrary to a more advanced (and therefore costly) tool called Firehorse, the free Stream API allows developers to fetch only 1% of the live stream. In other ways, considering all the tweets published at the present moment, one is able with the Stream API to get only 1% of the total tweets. But does this 1% sample is *representative* of all published content ? Studies [Morstatter et al., 2013] and [Morstatter et al., 2014] demonstrates that even if biases can be introduced, the sample data remains close and representative to the all stream.

In order to collect the data, the Twitter Stream API has been used with the keyword Ferguson between the August 10th and 27th. As a consequence, every tweet containing this word (not only hashtag) and present in the sample stream during that period is present in our dataset. In the end, our dataset contains approximately 12 million tweets.

[1] `http://inkdroid.org/journal/2014/08/30/a-ferguson-twitter-archive/`

2.2 Structure

The dataset contains approximately 12 million tweets and contains data from August 10th to August 27th. In this part we describe a tweet, a user and their structures.

2.2.1 Tweet structure

A **tweet** represents a message posted and its metadatas, like the publication date, its identification number, a media (photo / link) etc. Some metadatas are specific to Twitter :

* hashtag : represents a keyword in the message, written with the # symbol. It allows Twitter to classify tweets given their content.

* mention : represents a reference to another user in the message, written with the @ symbol. It is commonly used so people can talk to each other.

There are special tweets called *retweets*. They start by `RT @user` and allow a user to share a tweet of another user. A tweet with an important retweet count is considered as popular.

The detailed structure of a tweet is given on figure 2.1.

field	type	example
id	number	`498619822134280192`
publication date	date	`2014-08-11 00:00:04`
author id	number	`124010717`
text	number	`Please follow @AntonioFrench now!` `#Ferguson #MikeBrown http://t.co/---`
retweet count	number	`4`
is it a retweet ?	boolean	`False`
hashtags	string list	`#Ferguson, #MikeBrown`
mentions	string list	`@AntonioFrench`
links and medias	string list	`http://t.co/---`

Figure 2.1: Structure of a tweet

2.2.2 User structure

A **user** represents a person or an organization. It can be properly named with the real full name of the author or just use a nickname. In addition, a user has an identification nickname, usually short, that is used for mentions in a message, as described above. A user can follow and be followed by people. Contrary to social media like Facebook, here the links are unidirectional. If user A follows user B, user B may not follow user A. A user with an important followers count is considered as popular.

The detailed structure of a user is given on figure 2.2.

field	type	example
user id	number	`14090948`
name	string	`Antonio French`
nickname	string	`AntonioFrench`
registration date	date	`2008-03-06 19:51:29`
tweet count	number	`19372`
followers count	number	`119977`
friends count	number	`1373`

Figure 2.2: Structure of an user

2.3 Basic analysis

2.3.1 Frequencies

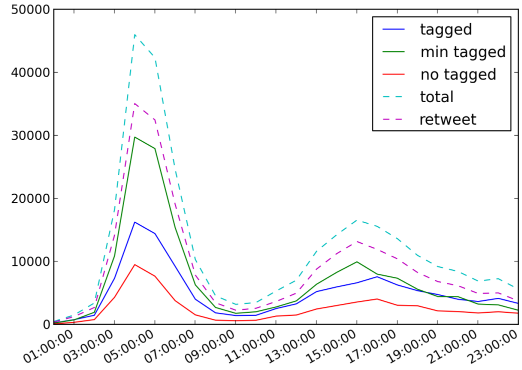
We compute the hour-frequency of tweets, that is to say the number of tweets published every hour for each day. The result is shown on figure 2.3. We distinguish several kinds of tweets :

* **tagged** : tweets containing a least one hashtag, <u>except</u> the two most popular that are `#Ferguson` and `#MikeBrown`.
* **min-tagged** : tweets containing a least one hashtag, <u>including</u> the two most popular that are `#Ferguson` and `#MikeBrown`.
* **no-tagged** : tweets with no hashtag at all
* **retweets** : only retweets, no simple tweets.
* **total** : all the tweets

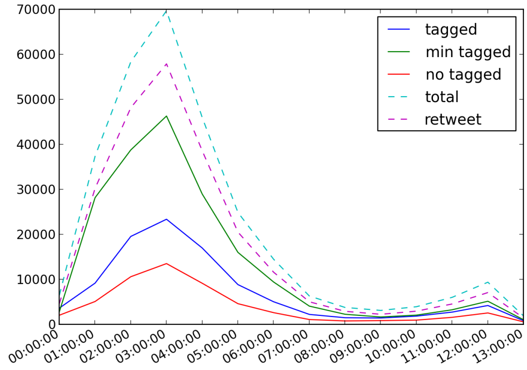
Firstly, one can see the frequencies distribution. We clearly have peaks of frequency during the nights, from midnight to the morning. This corresponds to the riots happening during the nights.

Moreover, one can notice the hashtags proportions. Approximately less than a half of the tweets (and sometimes only a third) contains non trivial hashtags (namely not `#Ferguson` or `#MikeBrown`). This raises the question of the relevance of studying hashtags in this situation.

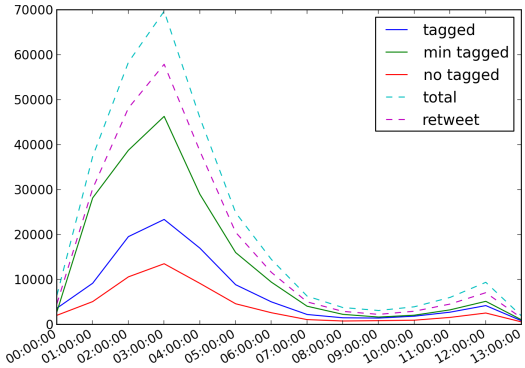
In addition, the proportion of retweets is very high each day. This means that most of the people just retweet relevant content instead of writing themselves.



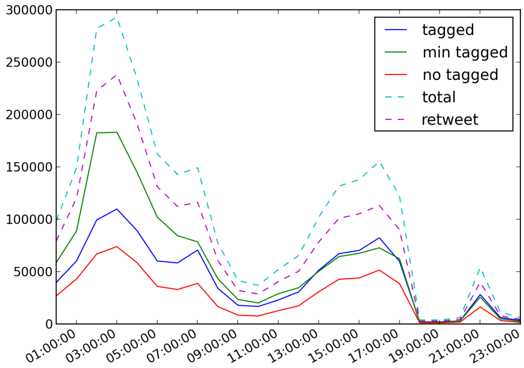
(a) August 11th



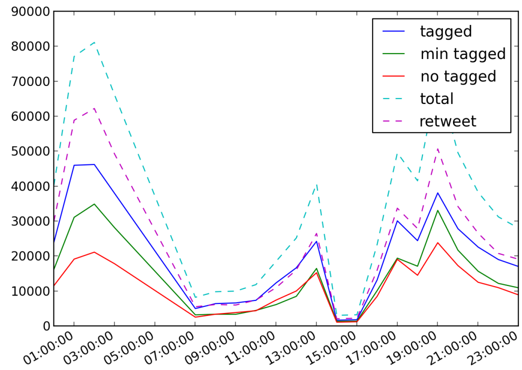
(b) August 12th



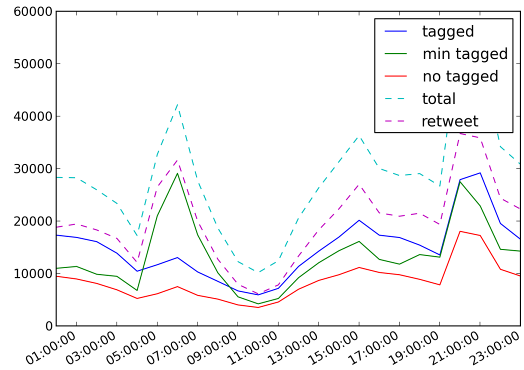
(c) August 13th



(d) August 14th



(e) August 15th



(f) August 16th

Figure 2.3: Tweets frequencies

2.3.2 Most retweeted users

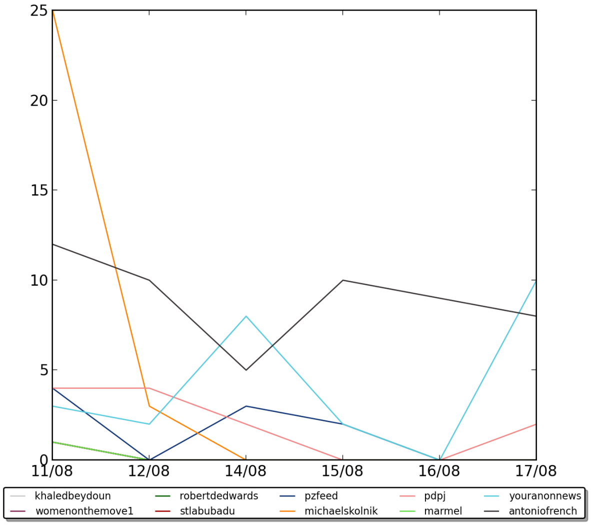
In this section we take a look at the most retweeted users. These are the ones for which the tweets appear among the top 100 retweets for a considered day.

The figure 2.4 shows the most retweeted users for four days. For each plot, one can see the most retweeted users of the corresponding day and their activity evolution on the other days. For example, on figure 2.4a, the user `michaelskolnik` has 25 tweets appearing in the top 100. But we see that on the August 12 he decreases to 3 tweets, and then he completely disappear. Some users like `antoniofrench` or `youranonnews` remains in the ranking almost for the six days.

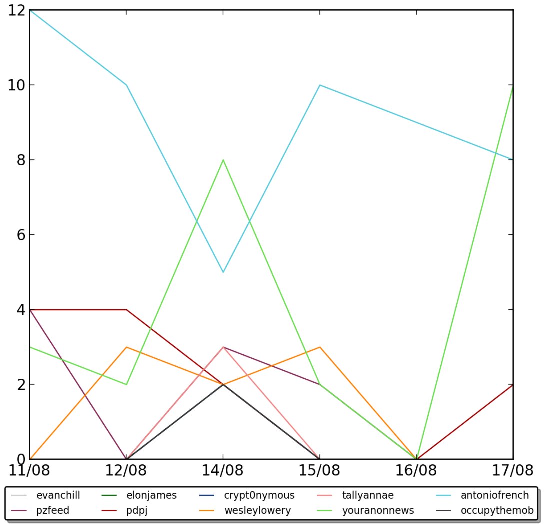
On figure 2.4c that corresponds to the August 16, we notice that the user `ryanjreilly` appears in the rank on the August 15 and begins more popular day by day. Moreover, some users like `koranaddo` or `michaelcalhoun` appear to be popular for only one day.

This highlights different kind of popular users :

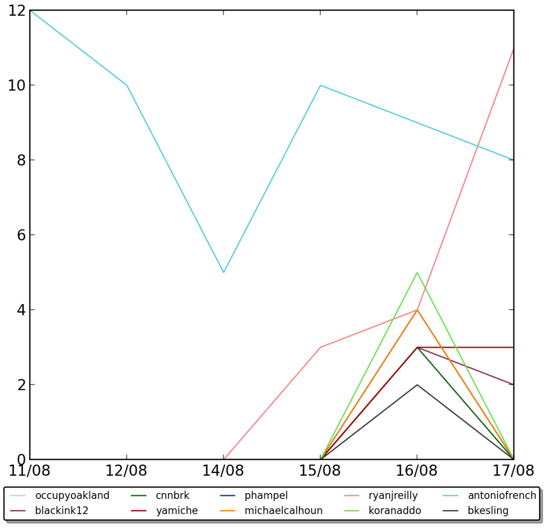
* the users very popular at the begining but who disappear after several days, and on the contrary the users who are not popular for some days but appear in the ranking a given moment ;
* the users who are only popular for a short moment, like a day ;
* the users that stay popular and retweeted a lot all through the protest.



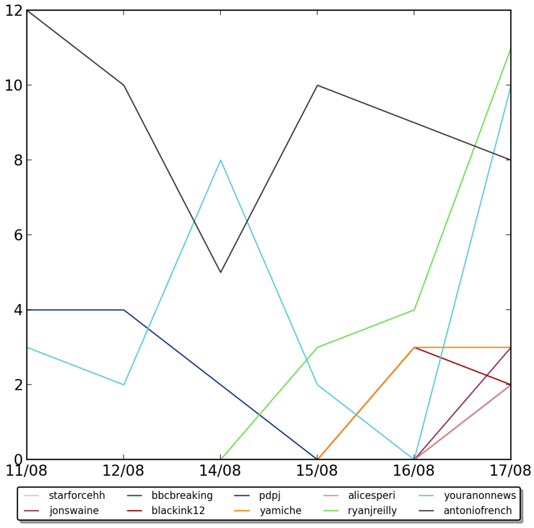
(a) August 11th



(b) August 14th



(c) August 16th



(d) August 17th

Figure 2.4: Most retweeted users on diffent days

Chapter 3

Topic modeling

3.1 Motivation

Before modeling users by the roles they play, we must define the support of reference, that is to say, the events or topics that will characterize their roles.

The simple approach is to consider a riot as a whole, but this has limits. Indeed, in this case we don't consider a subject but a time window, which is not very significant. For example, let's take two distinct events A and B happening the same day, and two users called *Alice* and *Bob*. Let's suppose *Alice* only tweets about the event A, and *Bob* only about the event B, and let's imagine they write positive things about their respective events. It would be wrong, only by considering the two events occur the same day, to conclude that *Alice* and *Bob* have the opinion. We must be sure they talk about the same things before comparing them.

In this chapter, we want to group the tweets by what they are talking abou, that is to say their underlying topics. Of course, it is not possible to automatically capture the meaning of the tweets and find topics that way, but we can try to clusters the tweets based on the word they contain.

After some preprocessing, we will try two methods in order to achieve this.

3.2 Preprocessing

Before analyzing the data, it's essential to preprocess it. We present the steps below.

3.2.1 Content preprocessing

```
RT @kharyp: How police treat white opencarry supporter vs an unarmed black
teen.  Wonder why?  #MikeBrown  #Ferguson  http://t.co/jG3hGJRAuC
```

1. **Cleaning :** removes all the non textual items :

   * mentions
   * hashtags
   * urls

   ```
   How police treat white opencarry supporter vs an unarmed black
   teen.  Wonder why?
   ```

2. **Tokenization :** breaks the sentence into a list of tokens, that can be words or symbols (punctuation). The tokens order is not considered anymore and a token can appear several times in the list.

   ```
   how, police, treat, white, opencarry, supporter, vs, an, unarmed,
   black, teen, ., wonder, why, ?
   ```

3. **Punctuation and Stop words removing :** keeps only the valuable tokens by removing symbols and too common words like *what*, *and*, *of* etc.

   ```
   police, treat, white, opencarry, supporter, vs, unarmed, black,
   teen, wonder
   ```

3.2.2 Tweet embedding

The goal is to characterize a tweet by the words it contains, by using for example the bag-of-words representation.

However, this method has its limits. Indeed, it does not consider how a word is common or not inside the whole corpus of tweets. For example, the word "Ferguson" is not very useful to characterize a tweet as it is a very common word in the corpus.

Instead, it is preferable to compute the TF-IDF, for term frequency – inverse document frequency, particularly used in text mining.

It is defined as follow :

$$tfidf = tf \times idf$$

where tf is the frequency of the word in the considered document (tweet) and

$$idf = log\frac{|D|}{|\{d_j|t_j \in d_j\}|}$$

where $D = \{d_j\}$ is the set of documents (tweets) and $|.|$ is defined as the size of the set.

In this way, for each word of a tweet, we take into consideration if this word is common in the dataset or not, and ajust its frequency accordingly. This method allows to bring out the uncommon words that may characterize a tweet.

We are now able to represent a tweet in the space of the vocabulary words $\mathcal{V}$:

$$\begin{array}{cc} & \begin{array}{cccc} word_1 & word_2 & \cdots & word_n \end{array} \\ tweet = & \begin{pmatrix} tfidf_1 & tfidf_2 & \cdots & tfidf_n \end{pmatrix} \end{array}$$

And all the tweets vectors generate the term matrix $\mathcal{M}$:

$$\begin{array}{cc} & \begin{array}{cccc} word_1 & word_2 & \cdots & word_n \end{array} \\ \begin{array}{c} tweet_1 \\ tweet_2 \\ \vdots \\ tweet_m \end{array} & \begin{pmatrix} tfidf_{1,1} & tfidf_{1,2} & \cdots & tfidf_{1,n} \\ tfidf_{2,1} & tfidf_{2,2} & \cdots & tfidf_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ tfidf_{m,1} & tfidf_{m,2} & \cdots & tfidf_{m,n} \end{pmatrix} \end{array}$$

3.3 Clustering using k-means

3.3.1 k-means algorithm

Given a set of documents $\mathcal{C} = \{d_1, d_2, \cdots, d_m\}$, we want to find k centers, or topics, $t_1, t_2, \cdots, t_k$ that minimizes the cost :

$$\sum_{i=1}^{n} \min_j ||d_i - t_j||_2^2$$

Algorithm 1 k-means algorithm

pick randomly k center $t_1, t_2, \cdots, t_k$
repeat
  assign each point d to its closest center t, in order to create k clusters
  compute the new centers by taking the mean of each cluster
until convergence

For a better initialization, some improved algorithms like k-means++ exist.

3.3.2 Topic modeling

After perfoming the k-means algorithm, we get two outputs :

* the **tweets clusters**, that is to say each tweet belongs to one cluster that correspond to a topic.
* the **topics**, that are the clusters centers. These are computed "mean" tweets from the tweets belonging to each cluster.

3.3.3 Experiment

We experiment k-means on two riot days : the August 11 (figure 3.1a) and the August 12 (figure 3.1b).

huge police crowd louis site st county live west killed
right police family nike liquor dollar goods sight store looted
right happening family people fire walmart shots department news looted
omg looting burning fire photo wow quiktrip quicktrip damn ground
florissant police stationed west burglarized walmart businesses officer w chambers
quicktrip burning people mo prayers n tonight go black need
feed ktvi stream rioting watch live updates coverage news scanner
right police looting riot people photo gas riots news dont
police fired area firing near walmart officer shots chambers scanner
via whats happening dellwood rioting wow damn killed shit looting

(a) August 11

right police riot get tear people via got told thats
protect police brutality watching people officers killing street moving allowed
steve aerial neck wooden walsh police tonight video shot pellet
leave good shot people resident media scene tonight right hope
woman police screams pregnant month slam im force nnht ground
force right police like scene looks another world sounds look
right police tear gas masks rubber bullets mist shooting stand
police area tv media trapped turned leave lot ordered told
right police dont people media scene cops go home happening
via police qt media outside stay protest right today happened

(b) August 12

Figure 3.1: Topics obtained with k-means

3.4 Latent Dirichlet Allocation

3.4.1 LDA algorithm

$$\begin{array}{ll} \text{Topics} & \mathcal{T} = \{t_1, t_2, \cdots, t_k\} \\ \text{Documents} & \mathcal{C} = \{d_1, d_2, \cdots, d_m\} \\ \text{Vocabulary} & \mathcal{V} = \{w_1, w_2, \cdots, w_n\} \end{array}$$

Let's consider that :

1. each document is a probabilistic mixture of several underlying topics ;
2. each topic is a probabilistic distribution of words ;

The topics are latent, hidden, as we do not observe them directly. They have to be infered from the words they are described with.

Dirichlet distribution

For three topics, the Dirichlet distribution for a document can be visualize in a triangle, as on figure 3.2. Each corner represent a topic. The closer a document is to a corner, the higher is the probability that the document belongs to the topic. For example, on the figure 3.2, the document belongs to the topic 2 with an higher probability than with the topics 1 and 3.

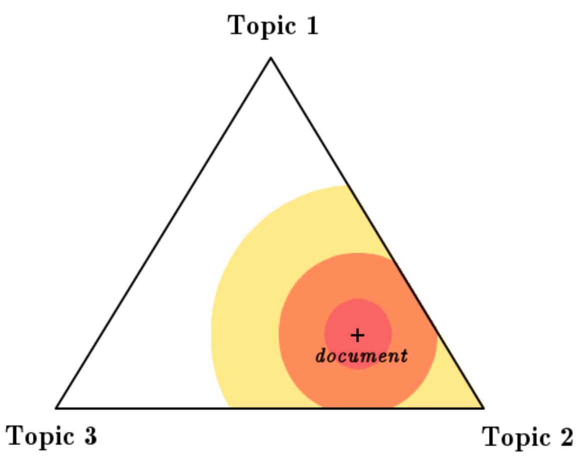


Figure 3.2: Hashtag graph

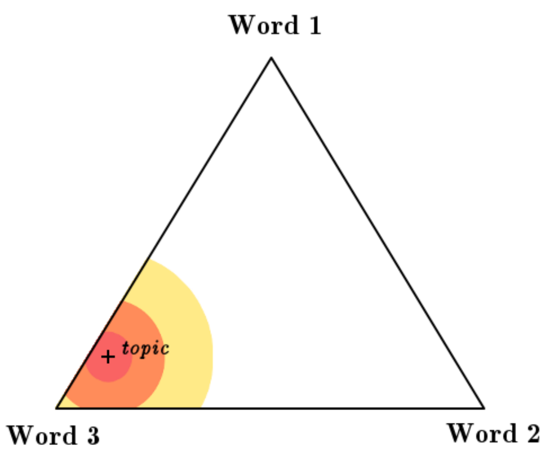


Figure 3.3: Hashtag graph

As figure 3.3 shows, we also have a Dirichlet distribution for the topic and words (here 3 words). In this simple example, the topic is mostly characterized by the word 3, and a bit more by the word 1 than the word 2.

3.4.2 Experiment

We experiment LDA on two riot days : the August 11 (figure 3.4a) and the August 12 (figure 3.4b).

racial profiling come shoot youth usa dont lessons survival black
right prayers language king unheard luther riot jr martin qt
praying officer police taco bell people swat news team pray
riots quiktrip today august watts ago began years burning tonight
quicktrip ground happening burning police looting gas tear wow scanner
photo burning quicktrip dogs community becomes police blame attention people
purge looting death mo people see guard via right police
police whats going nothing dellwood crazy difference changed pictures color
brown chaos looted police mike killed want looting shop oh
fired shots walmart live video fire police florissant looting quicktrip

(a) August 11

police gas tear officers stand riot masks protected mist live
n via historic tonight video protest like police st louis
nothing today get police home phones resident go happened thrown
police war missouri us shot riots city zone mo aerial
happening whats got going shit eyes need peace law news
media scene area pregnant leave told robin police williams thats
police people want dont black cops white watching street words
police violence shooting please journalists someone cars shut know anyone
right wow reporters gas tear photographers including bullets crowd rubber
jesus brown mike wow twitter pd obama time alex statement

(b) August 12

Figure 3.4: Topics obtained with Latent Dirichlet Allocation

3.5 Comparison of the two methods

Firstly, k-means is a *hard-clustering* method, that is to say one tweet belongs to only one cluster (ie one topic) and the clusters are disjoint.

On the contrary, LDA is a *soft-clustering* method. The algorithm represents a tweet as a mixture of topics. As a consequence, with that model, a tweet can belong to several topics with some probabilities.

Secondly, we can call into question the k-means approach essentialy because of the high-dimensionality of the vectors and their sparsity. Indeed, it is hard to evaluate if two vectors are close in a such voluminous space (for example with a vocabulary of 1000 words) especially as many components are equal to zero (as a tweet contains at most 140 characters). Such dimensional problems will also be discussed in section 6.2.

As a result, we choose to use the Latent Dirichlet Allocation method in the next of this project, as it seems to be the most realistic model.

Chapter 4

Language analysis and content polarity

4.1 Motivation

The main idea in this chapter is to characterize a user by what he writes in his tweets and in a general way we try to study the nature of the content shared by all the users during a riot.

Contrary to the previous chapter, we here tend to apply *supervised learning* methods, that is to say we want to classify the tweets given an already existing classification of content types.

In the first part, we will establish a content types classification and we will build a model to predict it. In the second part, we will use the media enclosed to the tweets in order to improve this model.

4.2 Characterizing tweets by their types

In this section, we classify tweets by their nature. We establish a simple type classification below.

4.2.1 Intuition

By taking a look at some tweets obtained on the 11th of August, we can underline several types of tweets and construct the following classification :

type	details	example
informative	The tweet contains informative text content about the happening riot. The URL link is ignored.	"#Ferguson quiktrip burning http://t.co/1Jm5ngZe62"
non informative	The tweet expresses feeling, an idea, a conviction, but no information about the happening riot.	"Prayers for #STL #Ferguson."
supportive	The tweet support, encourages directly the happening riot, the rioters or their ideas.	"Just spread the word we need #Justice" #ferguson #stlouis #MO #uniteblue #libcrib #p2 http://t.co/78QURveWvK"
not supportive	The tweet is objective about the situation or blames the riot, the rioters or their ideas.	"I am beyond saddened to see the looting in #Ferguson. This removes the focus from justice for #MikeBrown. Violence is never an answer."

The figure 4.1 shows more examples of manually classified tweets. The second column highlights semantical or syntaxical details that help to classify the given tweet. One can notice the types pairs appearing, mostly :

* informative / not supportive : tweets of interest as they are describing the riot situation in an objective way, for example eyewitnesses.
* non informative / supportive : useful tweets as their authors support the riot. It's interesting to consider the content in order to :
  - understand the rioters claims ;
  - observe if the "virtual violence" (language for example) of these users evolves ;
  - identify the main influencers.

The two others types are less interesting. Indeed, non informative / not supportive tweets does not contain any valuable information and informative / supportive tweets should be avoided as they may not contain objective information.

id	content	type
1	"A riot is the language of the unheard." Martin Luther King, Jr. #Ferguson http://t.co/OXfzgEcN1B	non informative not supportive
2	The Watts Riots began on August 11, 1965. 49 years ago today. #Ferguson http://t.co/gYgaI5WlP9	informative not supportive
3	QuickTrip is burning to the ground. #Ferguson http://t.co/9ykNn4a8ek	informative not supportive
4	Prayers for #STL #Ferguson.	non informative not supportive
5	The QuickTrip is now burning. #Ferguson (photo: @PDPJ) http://t.co/tJBeXQ6hzF	informative not supportive
6	"if you Retweet or Screenshot. Just spread the word we need #Justice" #ferguson #stlouis #MO #uniteblue #libcrib #p2 http://t.co/78QURveWvK	non informative supportive
7	Tear gas is being used in #Ferguson. Mayor tells CNN he has been receiving death threats. Armored vehicles roam the streets.	informative not supportive
8	PAY ATTENTION as "teen" becomes "man," "community" becomes "mob," and "murder" becomes "alleged shooting." #Ferguson #medialiteracy	non informative supportive
9	Happening now in #Ferguson http://t.co/KFuaOEZ4tf	informative not supportive
10	Nothing has changed. The ONLY difference? The pictures are in color. #Ferguson #FergusonShooting http://t.co/7p23qyDGwY	non informative supportive
11	#Ferguson quiktrip burning http://t.co/1Jm5ngZe62	informative not supportive
12	Men have pulled a truck up to the QT and are loading the ATM into the back. #Ferguson	informative not supportive
13	SWAT Team. Taco Bell. #Ferguson (photo:@kodacohen) http://t.co/Mo8CDVWx32	informative not supportive
14	Shots fired at Walmart. #Ferguson	informative not supportive
15	I am beyond saddened to see the looting in #Ferguson. This removes the focus from justice for #MikeBrown. Violence is never an answer.	informative ? not supportive

Figure 4.1: Examples of maunal labeling

4.2.2 Multinomial Naives Bayes Classifier

The model is the Multinomial Naive Bayes, a well-known probabilistic learning method. The Naive Bayes is a supervised method, which means it learns from a labeled dataset called training set. Once the model is trained, it can be used on unlabeled data. In order to measure the model accuracy, we run is on a test dataset.

Naives Bayes

Let $\mathcal{D} = \{d_i\}_{0 \leq i \leq n}$ be the set of n documents in the dataset.

For each document d_i, let $x_1, x_2, \cdots, x_m$ be its m features, such as

$$d_i = (x_1, x_2, \cdots, x_m)$$

Let's assume the features are independent.

The probability that the document d_i belongs to class C_k is given by :

$$P(C_k|d_i) = \frac{P(C_k)P(d_i|C_k)}{P(d_i)}$$

i.e.

$$P(C_k|x_1, \cdots, x_m) = \frac{P(C_k)\prod_{j=1}^{m} P(x_j|C_k)}{P(d_i)}$$

The predicted class $\hat{C}$ is then given by the class that has the maximum probability :

$$\hat{C} = \arg\max_k P(C_k|x_1, \cdots, x_m)$$

Word representation

In our case, a document can be represented as a bag of words, which is a set where the word frequency is considered.

For example, given a document d : `prayers,mike,brown,prayers,ferguson`

A bag-of-words representation would be :

prayers	mike	brown	ferguson
2	1	1	1

Considering several documents as a dataset, we define the vocabulary $\mathcal{V}$ as the set of all the words existing in the dataset.

Continuing on the previous example, if $\mathcal{V}$ is defined as {brown, ferguson, fire, mike, police, prayers, riots}, the document d can be represented as a count vector :

$$\begin{array}{cccccccc} & brown & ferguson & fire & mike & police & prayers & riots \\ d = & \big(\; 1 & 1 & 0 & 1 & 0 & 2 & 0 \; \big) \end{array}$$

This is a multinomial document representation.

Multinomial Naives Bayes

The Multinomial Naive Bayes is a common variant for text classification.

$$\boxed{P(x_j|C_k) = \frac{N_{k,j} + \alpha}{N_k + \alpha n}}$$

with

* $N_{k,j}$ the count of the feature x_j in the class C_k
* N_k the total count for all the features in C_k
* α is a smoothing parameter, such that $\alpha \leq 1$, to avoid a zero value in the product of the naives bayes equation.

Pair Naive Bayes

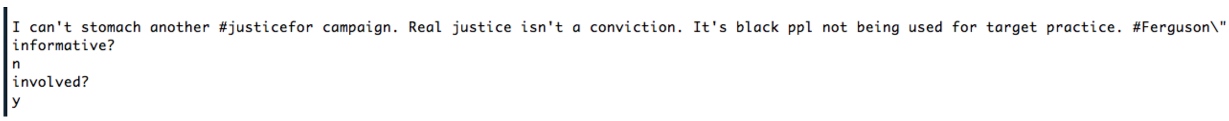
We will compare the previous model to another one that we call Pair Naive Bayes. The idea is to use the previous model not on the three classes in one time, but in two times. We first consider two classes, the class garbage and the two classes informative and supportive together, and we apply the model on it. We then perform our model on classes informative and surpportive, without considering the garbage class. We process here by pairs.

Training the model

In order to train our model, we manually label 500 tweets randomly choosen from the first riot (11 August). We annotate each document with two labels as follow :

informativness	involvement	
0	0	¬ informative & ¬ involved
0	1	¬ informative & involved
1	0	informative & ¬ involved
1	1	informative & involved

One can see on figure 4.2 the manual labeling process. The script shows a tweet message, and asks two *boolean* questions : Is the tweet informative ? Is the tweet involved ?



```
I can't stomach another #justicefor campaign. Real justice isn't a conviction. It's black ppl not being used for target practice. #Ferguson\"
informative?
n
involved?
y
```

Figure 4.2: Screenshot of the manual labeling process

Tweet type repartition on a manually classified 500 tweets sample

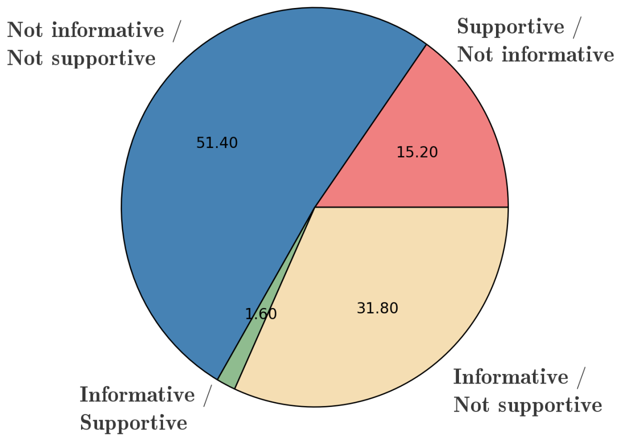


Figure 4.3: Tweets type repartition for the informative and involved features

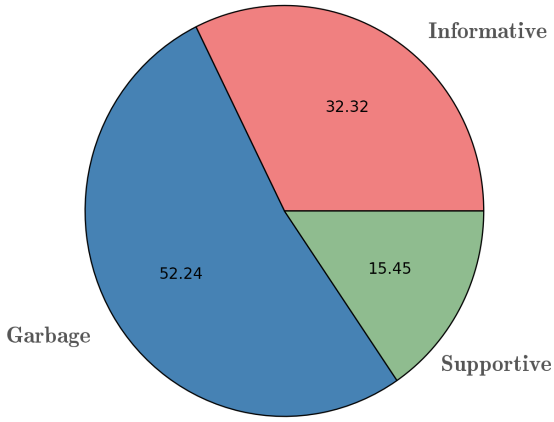


Figure 4.6: Proportions in dataset

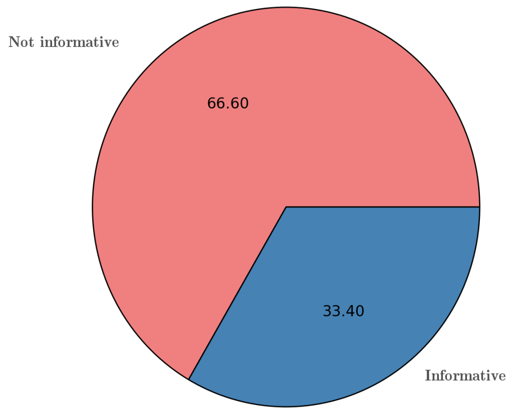


Figure 4.4: Tweets type repartition for the informative feature

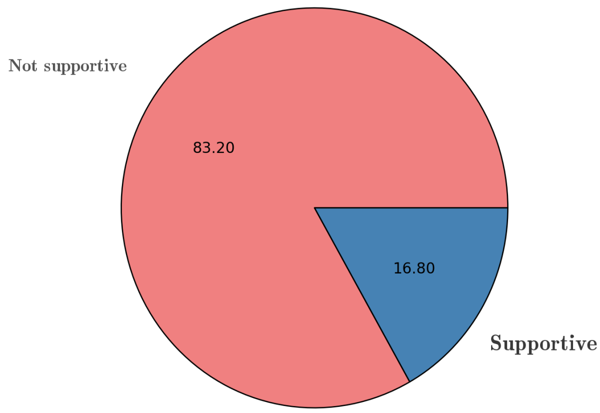


Figure 4.5: Tweets type repartition for the involvement feature

To visualize the repartition of the features informative and supportive, we plot the following pies.
Firstly, one can notice on the figure 4.3 how the types are divided :

* half of the tweets are neither informative nor supportive. This is what we can call the *garbage* category, as the concerned tweets are not worthy of interest.
* a very small part ($\sim 1.6\%$) is both informative and supportive. Considering the small percentage, we can consider this category is not significant.
* the remaining are either (exclusively) supportive or informative, with almost 2 times more informative tweets.

The figures 4.4 and 4.5 show the type repartition only for respectively the informative and supportive features, and lead to the following statements :

* more that half ($\sim 67\%$) of the tweets contain no informative content.
* surprisingly, only a bit more than a quarter ($\sim 17\%$) are supportive.

Testing

In order to test the model accuracy, we use the k-fold cross-validation technique with the F_1-score.

The easiest way to test a model is to split a labeled dataset in two parts : the training set s_1 (for example 80% of the whole dataset) and the testing set s_2 (20%). Once the model is trained with the set s_1, we run the model with the test set s_2, and compare the predicted labels with the true labels of s_2.

For a better trusted accuracy, it is common to use **k-fold cross-validation** method :

1. divide the dataset in k parts ;
2. use the first part as the testing set, and the union of the $k-1$ other parts as the training set ;
3. compute accuracy (in our case the F_1-score) ;
4. rotate such as each part is used once as the testing set and the others as training set ;
5. compute the average accuracy

For the accuracy, we use the F_1-score, given by :

$$F_1 = \frac{2 \cdot \text{true positive}}{2 \cdot \text{true positive} + \text{false negative} + \text{false positive}}$$

The test results are shown in figure 4.7, with $k = 5$. NB refers to Multinomial Naive Bayes, and PNB to Pair Multinomial Naive Bayes, as described before. We try to train the model with unigrams (tokens of single words) and bigrams (tokens of two words). The best results are obtained with unigrams with the NB model, as achieve a average accuracy of almost 65%. However, the score of class 2 remains bad (14.5%), so the model does not currently suit to classify our tweets. Two reasons appear :

1. it is harder to characterize class 2 by its content than the other classes ;
2. we do not have enough data to train the model.

	class #	0	1	2	average
	meaning	$\neg$ i & $\neg$ s	i & $\neg$ s	$\neg$ i & s	
NB	unigrams	76.1%	72.8%	14.5%	64.9%
NB	bigrams	74.3%	68.4%	4.2%	61.1%
PNB	unigrams	68.6%	70.0%	18.9%	60.8%
PNB	bigrams	62.5%	64.5%	10.1%	54.6%

Figure 4.7: Accuracies 5-fold cross-validation ; i = informative, s = supportive

4.3 Media analysis

As seen in the previous part, it is difficult to classifiy the supportive tweets with our training data. We try in this part to improve our model with the medias.

4.3.1 Motivation

One can notice on figure 4.8 that the tweets containing a media represent half of the total tweets, on average. As a result, it is important to take it into consideration and to analyze it.

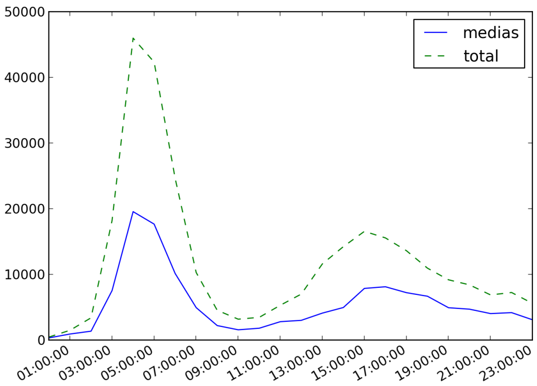


Figure 4.8: Medias proportion on August 11th

If we take a look at the tweets on the 11th of August labeled as supportive, we can extract the media on the figure 4.9. On the left, there is the domain names of the medias and on the right their descriptions.

The medias types are various : videos, images, blog articles, chatting window etc.

media domain name	description
youtube.com	Anonymous Operation video
cnn.com	video of Brown's mother "you took my son away from me"
vine.co	a man encouraging the protesters. "No fear. Keep going."
twimg.com	picture about racism
talk.ee	chat window
theobamacrat.com	blog article defending Mike Brown
twitter.com	tweet/photo of an activist (deleted since)
stltoday	article defending Mike Brown
youtube.com	video clip encouraging riots

Figure 4.9: Media in the training set labeled as supportive

4.3.2 Formulation

Let's define $\mathcal{M}_r = \{m\}$ the set of media m shared during the riot r. Each media is associated to n different tweet messages, without considering retweets.

Hypothesis : if the class of a media m is $C \in \{0, 1, 2\}$ then the associated tweet messages belongs also to this class.

4.3.3 Annoting the top 100 medias

We first consider the 100 medias the most retweeted on the 11th of the August, and we label it manually :

* **garbage** (class 0)
* **informative** (class 1)
* **biased** (class 2)

Figure 4.10: Media the most shared on the 11/08 night

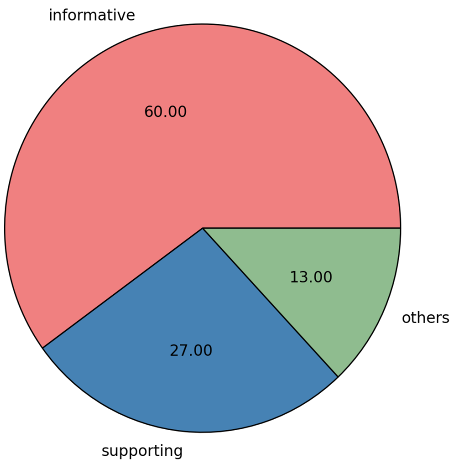


Figure 4.11: Media type repartition on the 11/08

The type repartition can be seen on the figure 4.11. More than a half of the media are informative, and a third is supportive. The media the most shared is shown on the figure 4.10.

4.3.4 Improving the classification model

Annoting only 100 medias enables to actually label several thousand tweets. We perform again our model testing, and we obtain the following results, on figure 4.12. The results are much better than before as we reach now an accuracy of 95.6% with the Pair Naive Bayes model. We will be able to use this model in the future to classify our tweets.

	class #	0	1	2	average
	meaning	¬ if & ¬ iv	if & ¬ iv	¬ if & iv	
NB	unigrams	84.9%	97.4%	85.9%	95.3%
	bigrams	80.4%	97.4%	78.5%	94.7%
PNB	unigrams	85.9%	97.6%	85.7%	95.6%
	bigrams	83.2%	97.5%	79.9%	95.1%

Figure 4.12: Accuracies 5-fold cross-validation ; if = informative, iv = involved

Given one of our Naive Bayes model, it is easy to get the most important features of each class. These are shown on figures 4.13 and 4.14.

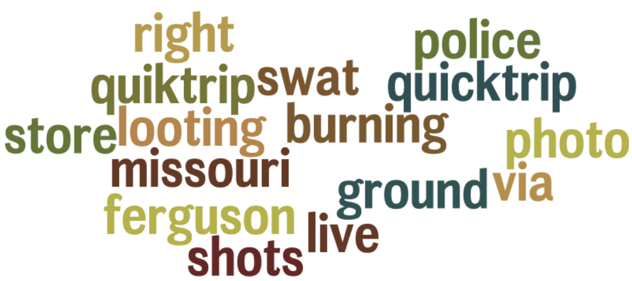


Figure 4.13: Most important features of the class 1 (informative)

Figure 4.14: Most important features of the class 2 (supportive)

One can see how the features of class 1 gather descriptive words, such as *looting*, *burning*, *shots* etc. The class 2 talks about the *hackers* and use different kind of words.

4.3.5 Media embedding and visualization

In this section we try to visualize the users given the media they published.
In order to achieve this, we represent a user by a media vector, in the same way we did in section 3.2.2 but now with medias.

$$\begin{array}{cc} & \begin{array}{cccc} media_1 & media_2 & \cdots & media_n \end{array} \\ user_1 = & \begin{pmatrix} tfidf_{1,1} & tfidf_{1,2} & \cdots & tfidf_{1,n} \end{pmatrix} \end{array}$$

Given the n most shared medias during a riot, each component i of the vector is the $tfidf$ frequency of the media i for the user.

Moreover, if the users posted a media that appears in the top 100 medias we labeled (section 3.2.2) as classified as *supportive*, we mark the user as “supportive”.

Eventually, we take the 1000 most active users, apply t-SNE transformation and plot it, as shown on figure . The black dots represent the “supportive” users, and the grey dots the others. One can notice on the figure that two clusters appear.

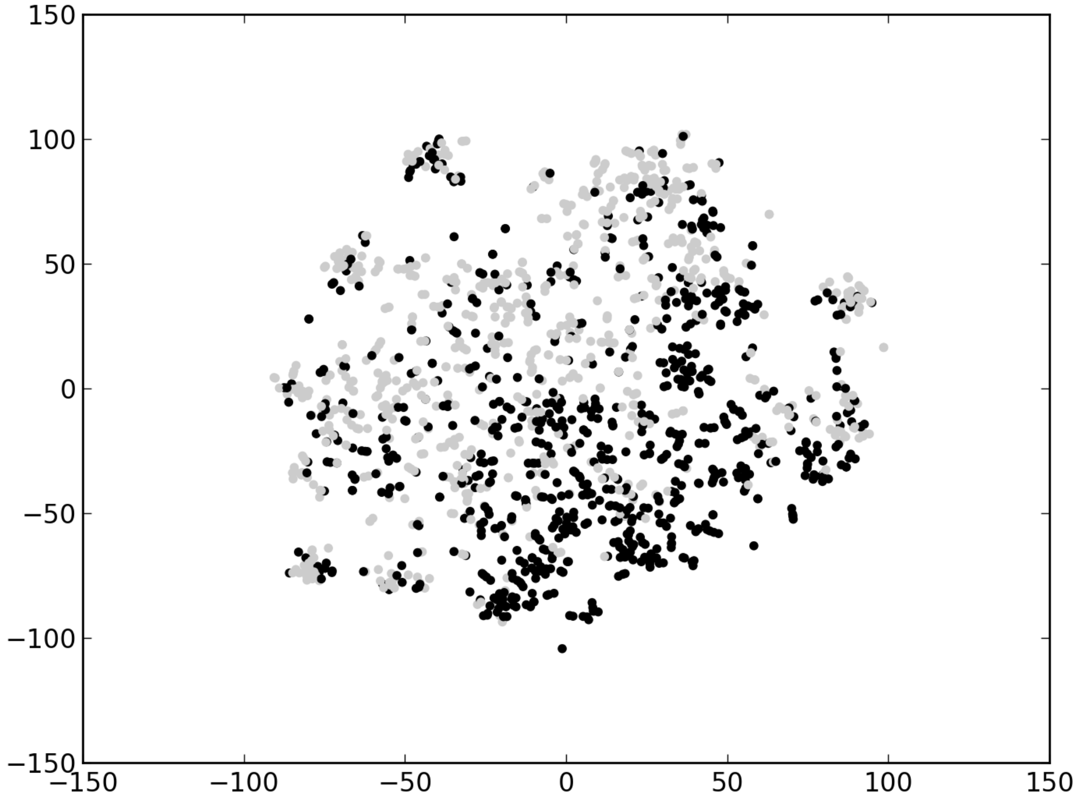


Figure 4.15: t-SNE transformation applied on the media vectors

4.4 Polarity score

As we saw, a tweet is either informative, supportive or garbage (neutral).
We now define the polarity score of a user considering a set of tweets.
Given a user u, we call the p_i the proportion of informative tweets in the set, and p_s the proportion of supportive tweets.
We define the polarity score of a user for a set of tweets as follow :

$$\mathcal{P}(u) = p_i - p_s$$

By definition, the polarity score is between -1 and 1. If the user is more supportive, his polarity will be negative. On the contrary, if he is more informative, his polarity will be positive.

Chapter 5

Graph analysis and influence network

5.1 Motivation and outline

In the past chapters we focused on the content analysis of the tweets. However, we have to keep in mind that we are working with data from a social network. As a consequence, it is important to study the underlying graphs interesting for our objectives.

In this chapter we start with computing several graphs in order to explore the dataset, then, we will focus on how to define influence in a network.

5.2 Hashtags graph

We start with analyzing the hashtags combinaisons. We construct a undirected graph such as :

* a **node** represents a hashtag
* an **edge** between nodes *a* and *b* exists if *a* and *b* are used together is a tweet. Our edges are weighted such as the higher the weight, the more used hashtag combinaison.

5.2.1 Computation

The algorithm 2 builds a graph of the hashtags combinaisons.

Algorithm 2 Create hashtags graph

```
input : tweets list, empty graph
output : graph
for each tweet in list do
  for each hashtag do
    add hashtag to list
    if node 'hashtag' does not exist in graph then
      add hashtag node to graph
    end if
    for each couple in 2-permutations of list elements do
      if edge 'couple' does not exist in graph then
        add couple edge to graph
      else
        increase weight of the couple edge in graph
      end if
    end for
  end for
end for
```

One can see the result on figure 5.1. On this graph, only the top 100 edges are shown, and the hashtag `#ferguson` as been removed, considered as trivial. The more visible the edge, the higher the weight.

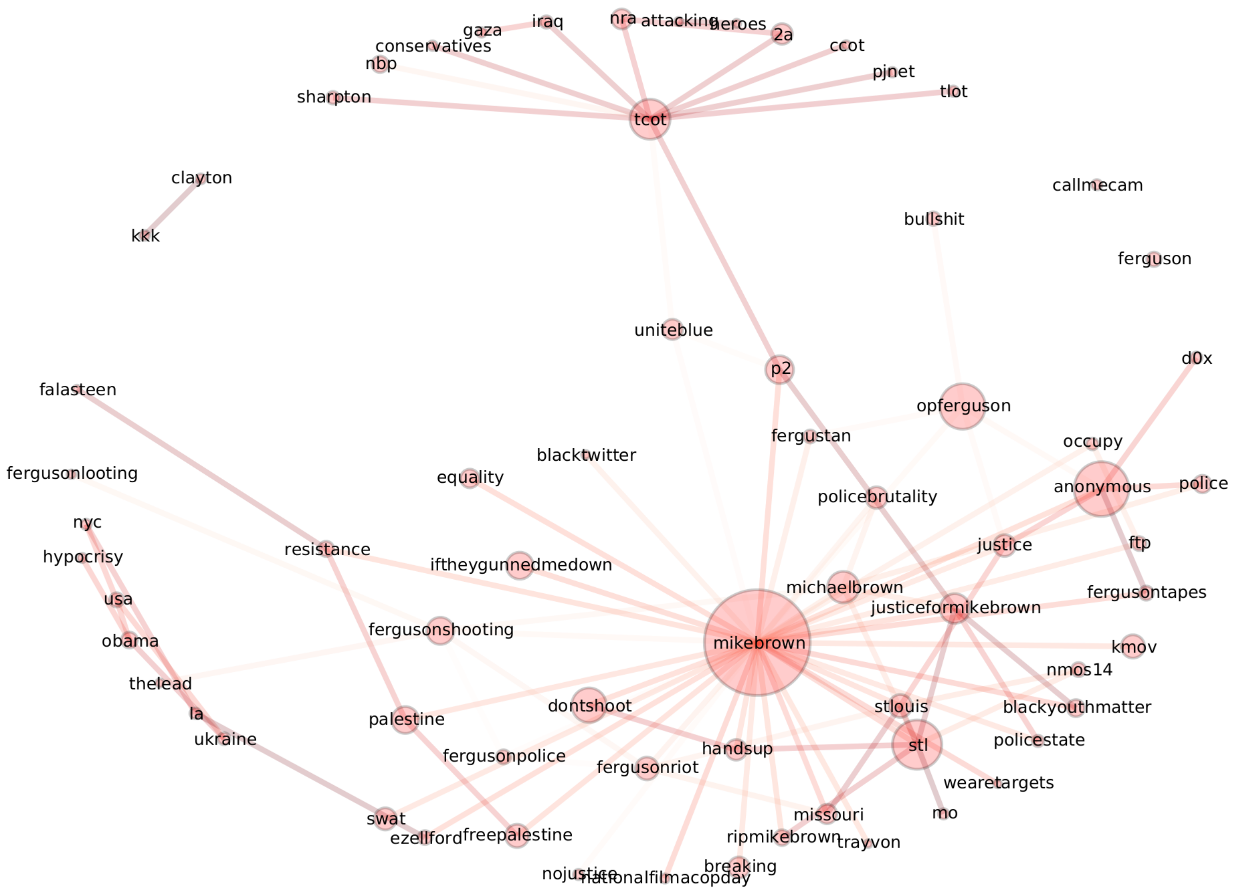


Figure 5.1: Hashtag graph for the August 13th.

5.2.2 Community detection

On the figure 5.1, there are evident groups of hashtags that can be considered as *communities*. In this section we try to detect automatically such communities in the graph.

In order to achieve this, we use the algorithm described in [Blondel et al., 2008], based on graph modularity optimization, and implemented in python by T. Aynaud[1].

The modularity[Newman, 2006] of a group can be defined as the observed number of edges in that group minus the expected number of edges if they were distributed randomly.

The number of communities to detect is a parameter of the algorithm. In our case, we set this parameter equals to 5.

The obtained graph is shown on figure 5.2.

[1]http://perso.crans.org/aynaud/communities/

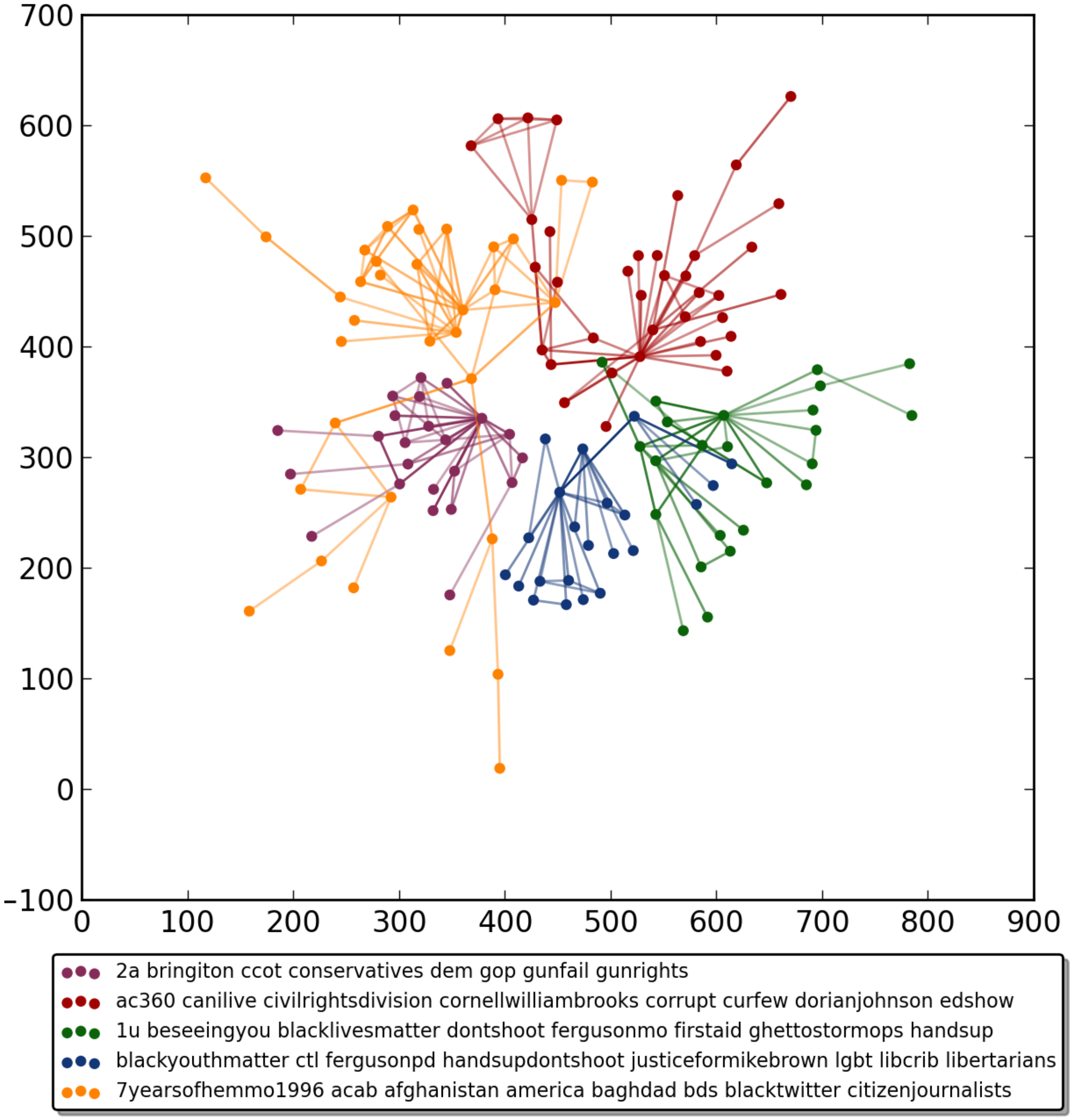


Figure 5.2: Hashtag graph

One can notice on the figure 5.2 we obtain distinct communities with different contents :

* **community #1** : political content related to the conservatives christians (`#ccot`), the conservatives, the democratic party (`#dem`), the republican party (`#gop`) etc.
* **community #2** : general content about black activist (Cornel W. Brooks), a friend of Mike Brown (Dorian Johnson), corruption etc.
* **community #3** and **community #4**: close communities with similar content related to defense of the blacks, with expressions like "black lives matter", "black youth matter", "hands up don't shoot", "dont't shoot" etc.
* **community #5** : content relative to the Middle East, like Afghanistan, Baghdad and Israel (`#bds`).

5.2.3 Contribution graph

We can wonder how users actually contribute to these hashtags communities. Do users frequently tweet with hastags of a same community or tweet in several hashtags communities ? For this purpose, we take the 100 users who tweeted the most in a community and we see if they tweeted in others communities. A big red dot represent a community with a given number, a small dot represent a user. A edge, in blue, links a user and a community. The opacity of the edge represents the proportion of the users tweets. The more the edge is dark blue, the more the user contributed to that community. The result is shown on figure 5.3.

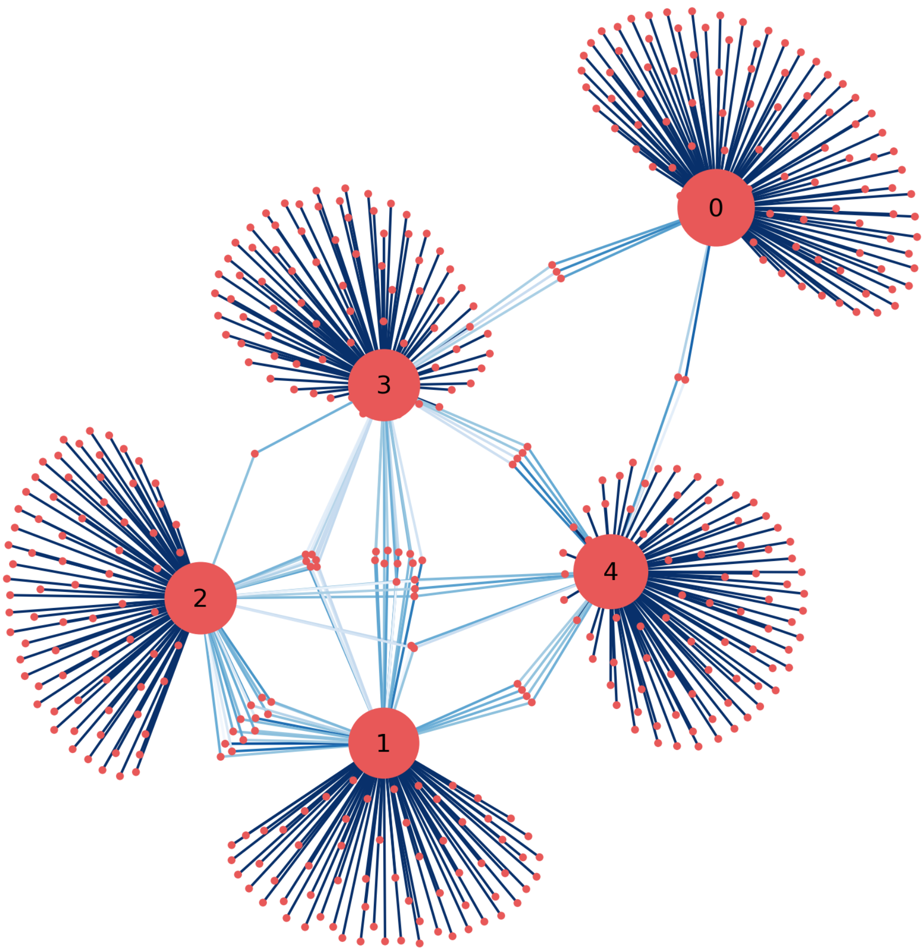


Figure 5.3: Hashtag graph

One can clearly see that most of the users only contributed in only one community. As a consequence, this means that hashtags induce both content and users communities.

5.3 Retweets graph

The other graph we can easily compute from our dataset is the mentions graph. As studied in section 2.3.1, a large proportion of tweets are retweets. As a consequence, we decide to keep only the retweet mentions in the graph, such as :

* a **node** represents a user
* an **directed edge** from node a to b exists if a retweets b. Our edges are weighted such as the higher the weight is, the more a has retweeted b.

5.3.1 Computation

The algorithm 3 builds a graph of the retweets.

Algorithm 3 Create retweets graph

```
input : retweets list, empty graph
output : graph
for each retweet in list do
   // we call a the the retweet author
   // and b the author of the original tweet
   if nodes a does not exist in graph then
      add hashtag node a
   end if
   if nodes b does not exist in graph then
      add hashtag node b
   end if
   if edge a → b does not exist in graph then
      create edge a → b
   else
      increase weight of edge a → b
   end if
end for
for each node in graph do
   keep the most important outgoing edge
end for
```

The result for the August 17th is shown on figure 5.4.

Figure 5.4: Retweet graph for the August 17th

On figure 5.4, we can clearly see the users that have been retweeted a lot : these are the big mass of points (the edges are not visible as they are too many points).

Moreover, the graph does not contain only one group but several, and these groups are connected each other. We obtain a *star* disposition, especially on the left, as small groups are connected to a big group.

This graph highlights the influence in the network. In big groups there are underlying very influent nodes. We can then say that the two big groups in the center of the graph represent the most influent users. In the same way, the nodes at the end are not influent in the network.

As a result, the aim is now to measure this influence, that is to say, that kind of position in the graph. We broach this point in the next section.

5.4 Influence in graph

5.4.1 Defining the influence

Degree Centrality

The degree centrality is the most simple way to characterize the influence of a node in a network. It is simply depends on the number of edges connected to the node. The more a node has edges, the more it is important in the network.

The degree centrality of node i is given by :

$$C_d(i) = \frac{d(i)}{n-1}$$

where d(i) is the degree of the node i and n is the number of nodes in the graph.

However, this measure has limits. Indeed, the node degree is local measure and it does not take into consideration the whole network structure.

Page Rank

The Page Rank[Page et al., 1999] was originally designed for web pages but it can be used for any kind of graph.

The main idea of the Page Rank is that an important node is pointed by other important nodes.

The Page Rank PR of a node i is given by :

$$PR(i) = \alpha \sum_{j \to i} \frac{PR(j)}{\mathsf{out}(j)} + \frac{1-\alpha}{n}$$

where :

* $\mathsf{out}(j)$ is the outdegree of j
* n is the number of nodes in the graph
* α is the *damping factor*, a constant parameter

We make an experiment with these two measures in the following part.

5.4.2 Experiment

We compute both In-degree centrality and Page Rank on the August 17th graph and we compare the top influent users lists obtained from these two measures. The result is shown on figure 5.5. The corresponding graph is shown on figure 5.6.

Rank	**In-Degree Centrality**	**Page Rank**
1	ryanjreilly	ryanjreilly
2	antoniofrench	huffingtonpost
3	youranonnews	youranonnews
4	fergusonunity	antoniofrench
5	youranonglobal	fergusonunity

Figure 5.5: Top influent users on the August 17th

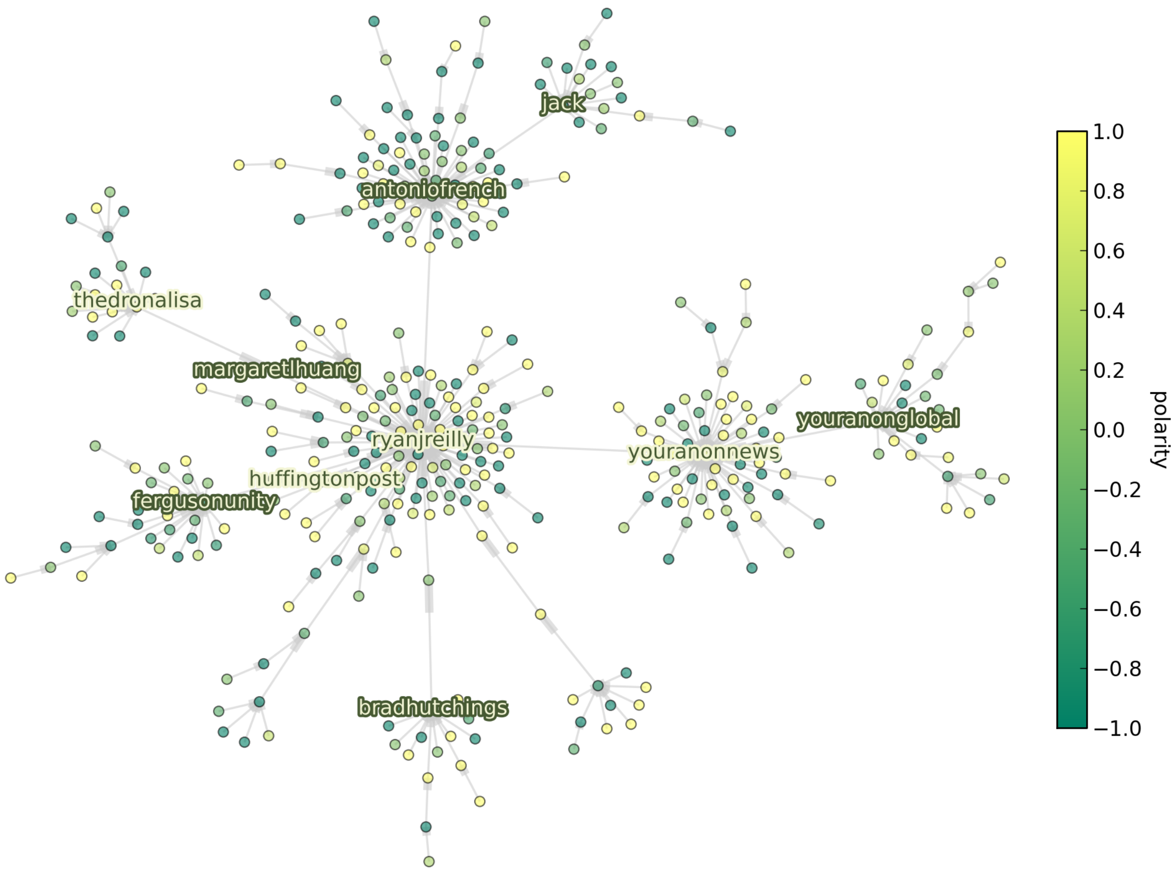


Figure 5.6: Influence graph on the August 17th

We can notice the `huffingtonpost` node does not appear in the top influent nodes with the degree centraliy measure. Indeed, it has a small degree but an incoming edge from the node `ryanjreilly`, so `huffingtonpost` is an influent node and only detected by the Page Rank measure.

The polarity is given by the green (= supportive) / yellow (= informative) colorbar. The top 10 influent users (given by Page Rank) are named on the graph. If a name is in dark, the corresponding user has a negative polarity. If a name is in light, the user has a positive (or null) polarity.

One can notice that there is no evident cluster between the informative and supportive users, they are all mixed together. We can however see groups with more supportive users, like the one on the top with the `antoniofrench` user.

5.5 Influence score

Given the experiment, we choose the Page Rank method to measure the influence of a user in the retweet graph. Moreover, we re-scale (function scale()) the influence score between 0 and 1. As a consequence, the most influence user will have a score of 1 and the non influent users will have a 0 score.

$$I(u) = \mathsf{scale}(PR(u))$$

Chapter 6

Role of a user

6.1 Definition

In this chapter we define and represent the role of a user during one or several riots based on the previous work. In the chapters 4 and 5, we introduced the *polarity* and *influence* scores for a user for a given set of tweets. Given the topics for a considered riot obtained in chapter 3, we can define :

* the *polarity vector* of a user for a riot as being the vector of the polarities of this user computed for the different topics belonging to the considered riot ;
* the *influence vector* of a user for a riot as being the vector of the influences of this user computed for the different topics belonging to the considered riot ;

We can now define the role of a user in a riot as follow :

> **Definition** - Role of a user for a riot
>
> Let u be a user and r a riot with n topics.
> Given the polarity vector $\mathcal{P}(u) = (p_{t_1}, p_{t_2}, \cdots, p_{t_n})$
> where $(p_{t_i})_{1 \leq i \leq n}$ is the polarity of u for the topic t_i,
> And given the influence vector $\mathcal{I}(u) = (i_{t_1}, i_{t_2}, \cdots, i_{t_n})$
> where $(i_{t_i})_{1 \leq i \leq n}$ is the influence of u for the topic t_i,
>
> We can define the *role* $\mathcal{R}(u)$ of u as being the concatenation of $\mathcal{P}(u)$ and $\mathcal{I}(u)$:
>
> $$\mathcal{R}(u) = (\mathcal{P}(u)|\mathcal{I}(u))$$

In a similar way, we can easily define the role of a user for m riots by concatenation :

Definition - Role of a user for m riots

Let u be a user and let $s = \{r_i\}_{1 \leq i \leq m}$ a set of m riots with n topics each.
Given the polarity vector $\mathcal{P}_{r_i}(u) = (p_{t_1}, p_{t_2}, \cdots, p_{t_n})$ of u for the riot r_i and in the same way $\mathcal{I}_{r_i}(u) = (i_{t_1}, i_{t_2}, \cdots, i_{t_n})$ its influence vector, we can define the new polarity and influence vectors for u for the set s as follow :

$$\hat{\mathcal{P}}(u) = (\mathcal{P}_{r_1}|\mathcal{P}_{r_2}|\cdots|\mathcal{P}_{r_m})$$

$$\hat{\mathcal{I}}(u) = (\mathcal{I}_{r_1}|\mathcal{I}_{r_2}|\cdots|\mathcal{I}_{r_m})$$

and the role of u for s as :

$$\hat{\mathcal{R}}(u) = (\hat{\mathcal{P}}(u)|\hat{\mathcal{I}}(u))$$

6.2 Dimension reduction

6.2.1 High-dimensional vectors

The vectorial space created is currently $20 \cdot m$ dimensional. This can be considered as high-dimensional and as a consequence causing trouble.

Firstly, because of the **curse of dimensionality**. In high dimensions, the space is big and sparse. As a consequence, it becomes difficult to evaluate if two vectors are similar/close or not.

Secondly, it is interesting to reduce the dimensions for **visualization** purpose.

6.2.2 Principal Component Analysis

Principal Component Analysis is a well-known statistical method to reduce dimensions of a dataset. It converts a set of correlated components to a new set of uncorrelated ones called principal components. Each principal component explains a percentage of variance such as the first components have the highest variances.

6.2.3 t-SNE

We choose to compare PCA with another dimension reduction algorithm more designed for visualization called t-SNE[Van der Maaten and Hinton, 2008].

6.3 Unified model

In this section we describe the unified model of the project. All the steps are represented on the figure **??**.

1. Compute the topics with Latent Dirichlet Allocation. (section 3.4)
2. For each topic, classify the tweets with Naive Bayes and compute a polarity score for each user. (section 4.3.4)
   $\rightarrow$ polarity vector
3. For each topic, generate a retweet graph and compute the Page Rank for each user. (section 5.4.1)
   $\rightarrow$ influence vector
4. Concatenate the polarity and influence vectors
5. Reduce the vectors to d dimensions with PCA or t-SNE, for example $d = 2$ for visualization purpose.
   $\rightarrow$ role vector

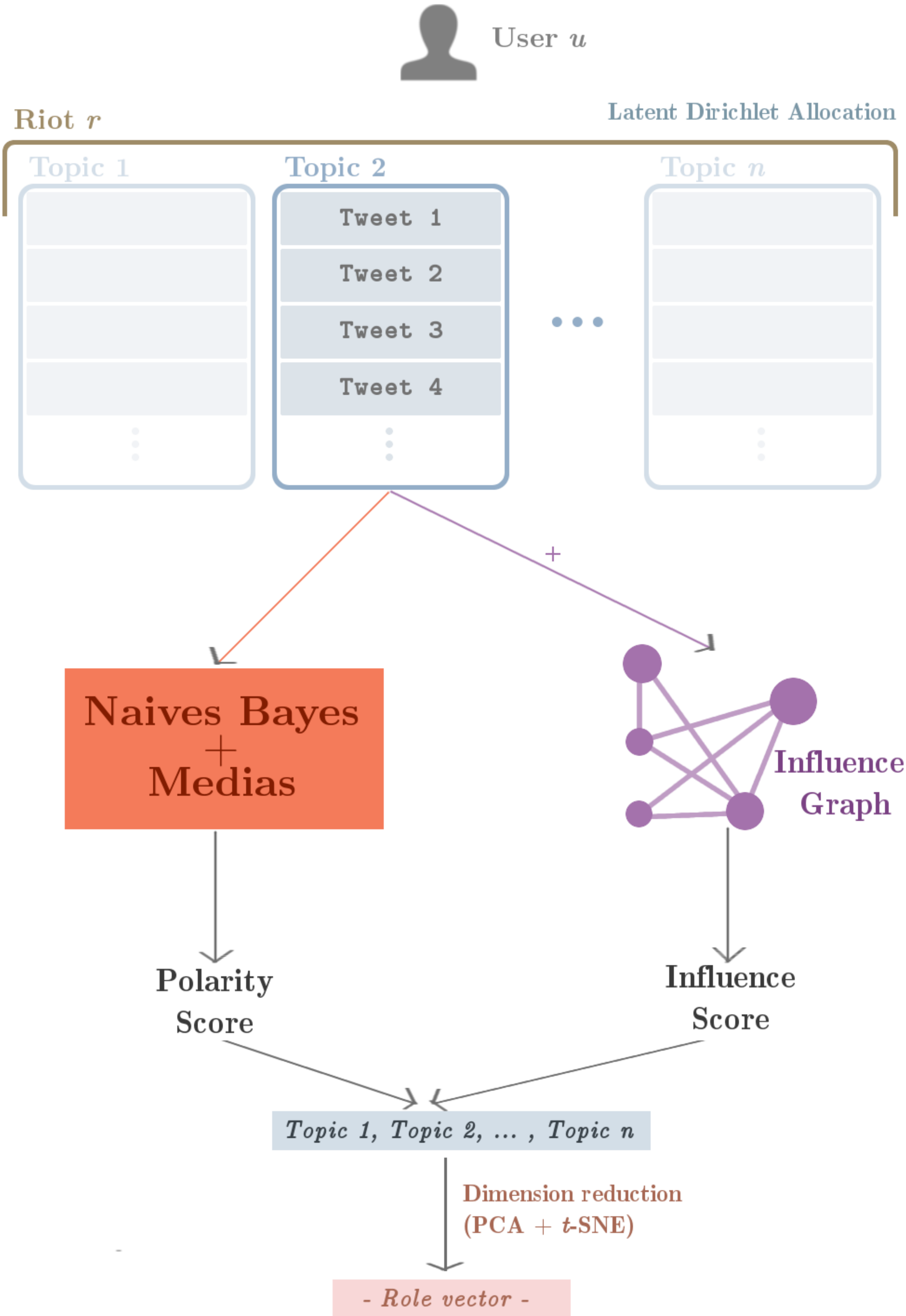


Figure 6.1: Unified model for the role vector computation
??

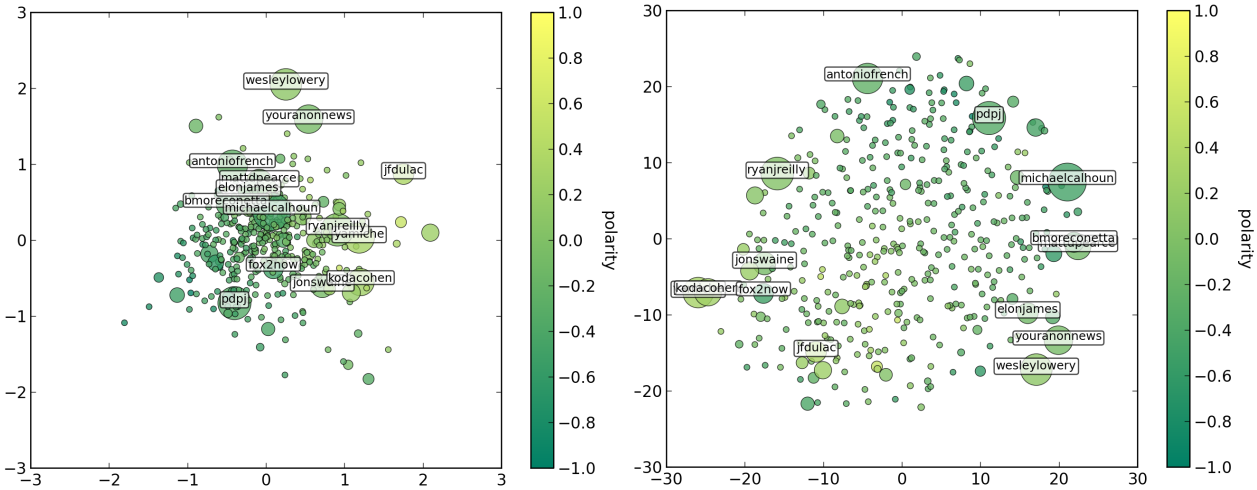
6.4 Experiments

We conduct two experiments. First, we compute roles for single riots and then in the next part for two riots.

6.4.1 Roles during a riot

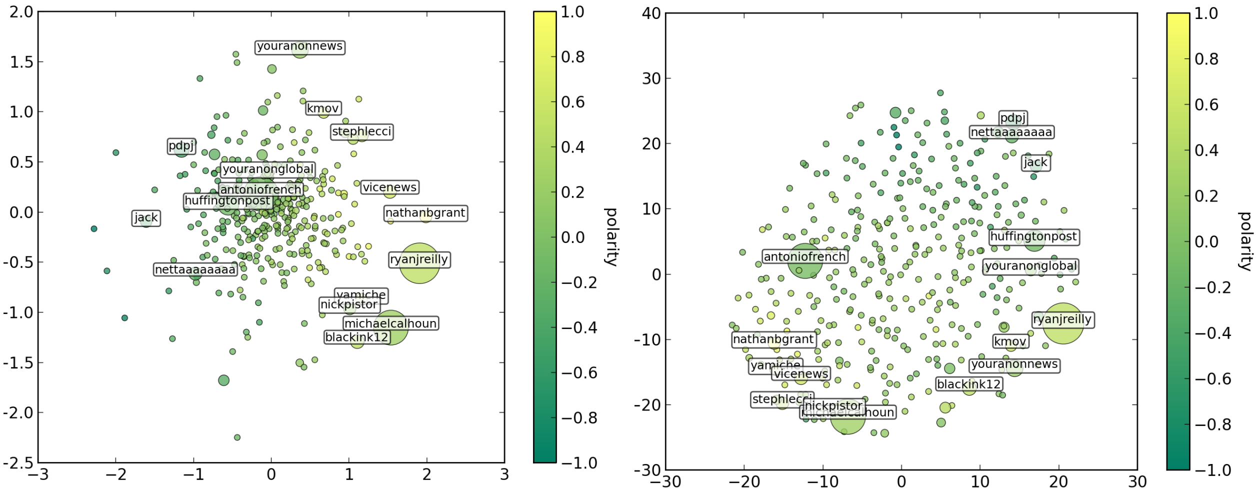
On figure 6.2, one can see the roles plotted after PCA and t-SNE reduction. The polarity is indicated with the color scale, and the influence is proportional to the dot sizes.

On both figures, there is a clear separation between the informative users (light green dots) and the supportive users (dark green dots). Moreover, we can notice that the most influent users (big dots) are not grouped together but mixed with the others. The same results are obtained on figure 6.3



(a) PCA (b) t-SNE

Figure 6.2: Visualization for the roles on the August 14



(a) PCA (b) t-SNE

Figure 6.3: Visualization for the roles on the August 17

We now put aside the concepts of influence and polarity and just analyze the disposition of the users on the plots. On figure 6.2a and 6.2b, the users `ryanjreilly`, `yamiche`, `kodacohen` and `fox2now` are close together. By taking a look at their profile description, it turns out they are all journalists or news medias.

In the same way, even if it is still compacted on the PCA plot, we notice that black personalities (i.e. people that may clearly defend black people in the Ferguson protests) like `antoniofrench`, `elonjames` and `wesleylowery`, as much as `youranonnews`, the account of the Anonymous hackers who also defended black people, appear in a different group on the plots (especially on the t-SNE plot).

However, on the figure 6.3, we notice differences in the disposition between PCA and t-SNE. Indeed, on figure 6.3a, the users `youranonnews` and `ryanjreilly` are almost on the opposite, whereas they are very close on figure 6.3b.

We also perform the k-Nearest-Neighbors, an algorithm that returns the k closest neighbors of a query point, with the Euclidean distance on the role vectors obtained from t-SNE. The results are shown on figure 6.4. On 6.4a, the query user is an activist hacker group, and we can see that the first and third users returns by the algorithm have described themselves as activists. In the same way, on the August 17, for the query `jonswaine` (journalist), one can see on figure 6.4b that there are five journalists or news related accounts and four of them are among the closest neighbors.

Rank	**user**	**description**
1	edreggi	actor/activist
2	2chainzlyrics	/
3	kathrynbruscobk	author/activist
4	robertloerzel	author/journalist
5	nickpistor	author/reporter
6	fallongreen15	/
7	roricomics	author
8	khatumofestival	/
9	kim_sutherland	/
10	michaelcalhoun	journalist

(a) August 14 with query user “youranonnews”

Rank	**user**	**description**
1	chrishayestv	journalist
2	plussone	activist
3	laureldavilacpa	radio commentator
4	twitchyteam	news software
5	michaelcalhoun	journalist
6	fallongreen15	/
7	_nealanae	/
8	wesknuckle	/
9	2chainzlyrics	/
10	nickpistor	journalist

(b) August 17 with query user “jonswaine”

Figure 6.4: Users returned with a k-NN search on the t-SNE values for a specified

6.4.2 Role similarity between 2 riots

We finally run our model considering two riots, the August 14 and 17. The t-SNE plot is shown on figure 6.5. Again, some users with same *function* are mapped close to each other, as for example `pzfeed` and `ryanjreilly`. However, we can also notice that on this plot the news media `fox2now` is mapped suprinsingly close to the hackers account `youranonglobal`.

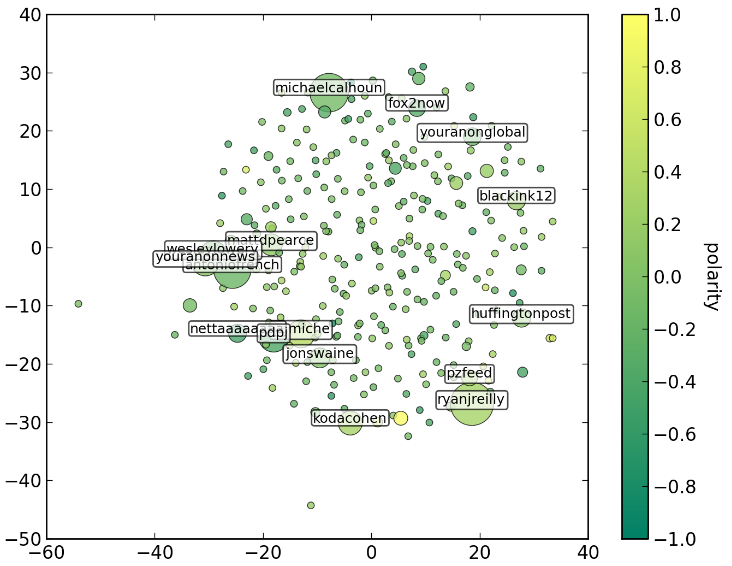


Figure 6.5: t-SNE plot of the roles on the August 14th and 17th

On figure 6.6, we perform k-NN algorithm on the data with query "pzfeed" (a news media). Excepting the first result that is an activist, the research only returns news-related users, what confirm our model groups quite well the users given their roles.

Rank	**user**	**description**
1	soulrevision	activist
2	copwatchnews	activists news
3	breaking3zero	news website
4	allthenewsisnow	news website
5	ryanjreilly	journalist
6	realtimehack	/
7	haiku_rs	/
8	washingtonpost	news website
9	monaeltahawy	author/journal columnist
10	youranoncentral	hackers account

Figure 6.6: Users returned with a k-NN search for the query user "pzfeed"

Chapter 7

Related work

Crisis events like riots have been a topic of study over the past few years in computer science with the progressive use of social media in the whole world. These studies lead to a new field called *crisis informatics*.

[Vieweg et al., 2015] analyzes data from 26 crisis events of several types (floods, earthquakes, wildfires, shootings, bombings etc.) and makes a transversal study of the information types and sources shared on Twitter. The study shows that :

1. Even if such events happen regularly, each crisis remains unique, even for two disasters of the same type occuring in the same country.
2. Human-induced crises (riots, shootings..) tend to be more similar to each other than to natural disasters.

[Imran et al., 2013] descibes a method for processing messages from social media in order to extract useful information. The proposed system uses machine learning methods like Naive Bayes to classify relevant messages and obtains good accuracy results.

[Mendoza et al., 2010] studies Twitter messages after the 2010 Chile earthquake. Among other things, this paper analyzes the propagation of false and true rumors and shows that false rumors can be detected. Indeed, these messages tend to be more called into question, and as a consequence machine learning can target them.

Finally, [Imran et al., 2014] is a state of the art about processing social media messages during crisis events. It describes all the steps, which are in the order : data acquisition, preprocessing, event detection, event tracking, clustering / classification, information extraction, summarization, semantic enrichment and ontologies. For each section, the paper describes the existing techniques, their results, and references the main relevant papers.

About the 2011 London riots, [Procter et al., 2013] analyzes by what kind of actor (media, celebrity, activist etc.) was used Twitter during that period. The paper also focuses on rumors, how they spread during the riots and how they can be called into question.

Finally, [Compton et al., 2013] tries to forecast social unrest by applying textual and geographic filters on the new messages published on Twitter.

Chapter 8

Conclusion

8.1 Summary

In this project we have introduced a framework for analyzing roles of users during a riot. The dataset contains messages from Twitter during the 2014 US Ferguson protests.

In chapter 3, we first perform some preprocessing on the data. Then, we extract topics from a riot by comparing two techniques : k-means and Latent Dirichlet Allocation. After experimenting, we choose to select the second method.

In chapter 4, we focus on the content of the tweets. After having established a classification of types, we train and test a Naive Bayes classifier to predict if a tweet is either supportive or informative about the riot. We also study the medias shared by users in order to improve our classifier and in the end we define and compute user polarity score.

In chapter 5, we perform graph analysis on the hashtags and users and we visualize these networks through examples. We also define and compute user influence score with degree centrality and Page Rank. We choose to keep for the next chapter the Page Rank method, as it is more realistic.

Finally, we define the role of a user during a riot (or several) as being the feature vector of its polarity and influence scores for each topic. We perform dimension reduction with PCA and t-SNE for visualization purpose and neighbors research. Through experiments, we show how users with the same job (for example journalists or activists) are grouped together.

8.2 Future work

In this section we identify some future work tracks to improve our framework.

Firstly, it is possible to perform more in-depth Natural Language Processing analysis in the begining. Indeed, it would be interesting to add a lemming or stemming step in section 3.2. These processing would allow to consider for example words as *riot* and *riots* to be represented as the same, as much as for the words *looting* and *looted*. In addition to reducing the vocabulary size, it may improve the model accuracy in section 4.2.2. We can also think about create a lexicon specific to riots in order to detect and classify better informative and supportive tweets.

Secondly, our model remains quite simple, as it only considers two features : the *influence* and the *polarity*. It may also be a good track to define new features to characterize the role of a user. For example, it would be possible to use the hashtags communities found in 5.2.2.

Finally, it would be necessary to perform more exhaustive tests on our framework by playing with the different parameters. For example, it would be interesting to try with a different number of topics or a different vocabulary size. Moreover, tests on more riots days should be performed to identify strengths and weaknesses of our model.

Bibliography

[Blondel et al., 2008] Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008). Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008.

[Compton et al., 2013] Compton, R., Lee, C., Lu, T.-C., De Silva, L., and Macy, M. (2013). Detecting future social unrest in unprocessed twitter data:"emerging phenomena and big data". In *Intelligence and Security Informatics (ISI), 2013 IEEE International Conference On*, pages 56–60. IEEE.

[Imran et al., 2014] Imran, M., Castillo, C., Diaz, F., and Vieweg, S. (2014). Processing social media messages in mass emergency: A survey. *arXiv preprint arXiv:1407.7071.*

[Imran et al., 2013] Imran, M., Elbassuoni, S. M., Castillo, C., Diaz, F., and Meier, P. (2013). Extracting information nuggets from disaster-related messages in social media. *Proc. of ISCRAM, Baden-Baden, Germany.*

[Mendoza et al., 2010] Mendoza, M., Poblete, B., and Castillo, C. (2010). Twitter under crisis: Can we trust what we rt? In *Proceedings of the first workshop on social media analytics*, pages 71–79. ACM.

[Morstatter et al., 2014] Morstatter, F., Pfeffer, J., and Liu, H. (2014). When is it biased?: assessing the representativeness of twitter's streaming api. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, pages 555–556. International World Wide Web Conferences Steering Committee.

[Morstatter et al., 2013] Morstatter, F., Pfeffer, J., Liu, H., and Carley, K. M. (2013). Is the sample good enough? comparing data from twitter's streaming API with twitter's firehose. *CoRR*, abs/1306.5204.

[Newman, 2006] Newman, M. E. (2006). Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582.

[Page et al., 1999] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web.

[Procter et al., 2013] Procter, R., Vis, F., and Voss, A. (2013). Reading the riots on twitter: methodological innovation for the analysis of big data. *International journal of social research methodology*, 16(3):197–214.

[Van der Maaten and Hinton, 2008] Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(2579-2605):85.

[Vieweg et al., 2015] Vieweg, S., Olteanu, A., and Castillo, C. (2015). What to expect when the unexpected happens: Social media communications across crises. *Proceedings of CSCW 2015 (forthcoming).*